A Subspace Learning Approach to High-Dimensional Matrix Decomposition with Efficient Information Sampling

Mostafa Rahmani, *Student Member, IEEE* and George K. Atia, *Member, IEEE*

***Abstract*—This paper is concerned with the problem of low-rank plus sparse matrix decomposition for big data. Conventional algorithms for matrix decomposition use the entire data to extract the low-rank and sparse components, and are based on optimization problems that scale with the dimension of the data, which limit their scalability. Furthermore, the existing randomized approaches mostly rely on uniform random sampling, which can be quite inefficient for many real world data matrices that exhibit additional structures (e.g. clustering). In this paper, a scalable subspace-pursuit approach that transforms the decomposition problem to a subspace learning problem is proposed. The decomposition is carried out using a small data sketch formed from sampled columns/rows. Even when the data is sampled uniformly at random, it is shown that the sufficient number of sampled columns/rows is roughly $\mathcal{O}(r\mu)$, where μ is the coherency parameter and r the rank of the low-rank component. In addition, efficient sampling algorithms are proposed to address the problem of column/row sampling from structured data. The proposed sampling algorithms can be independently used for feature selection from high-dimensional data. The proposed approach is amenable to online implementation and an online scheme is proposed.**

***Index Terms*—Low-Rank Matrix, Subspace Learning, Big Data, Matrix Decomposition, Column Sampling, Sketching**

I. Introduction

Suppose we are given a data matrix $\mathbf{D} \in \mathbb{R}^{N_1 \times N_2}$, which can be expressed as

$$\mathbf{D} = \mathbf{L} + \mathbf{S}, \tag{1}$$

where $\mathbf{L}$ is a low rank (LR) and $\mathbf{S}$ is a sparse matrix with arbitrary unknown support, whose entries can have arbitrarily large magnitude. Many important applications in which the data under study can be naturally modeled using (1) were discussed in [1]. The cutting-edge Principal Component Pursuit approach developed in [1], [2], directly decomposes $\mathbf{D}$ into its LR and sparse components by solving the convex program

$$\begin{aligned} \min_{\dot{\mathbf{L}},\dot{\mathbf{S}}} \quad & \lambda\|\dot{\mathbf{S}}\|_1 + \|\dot{\mathbf{L}}\|_* \\ \text{subject to} \quad & \dot{\mathbf{L}} + \dot{\mathbf{S}} = \mathbf{D} \end{aligned} \tag{2}$$

where $\|.\|_1$ is the ℓ_1-norm, $\|.\|_*$ is the nuclear norm and λ determines the trade-off between the sparse and LR components [2]. The convex program (2) can precisely recover both the LR and sparse components if the columns and rows subspace of $\mathbf{L}$ are sufficiently incoherent with the standard basis and the non-zero elements of $\mathbf{S}$ are sufficiently diffused [2]. Although the problem in (2) is convex, its computational complexity is intolerable with large volumes of high-dimensional data. Even the efficient iterative algorithms proposed in [3], [4] have prohibitive computational and memory requirements in high-dimensional settings.

Contributions: This paper proposes a new randomized decomposition approach, which extracts the LR component in two consecutive steps. First, the column-space (CS) of $\mathbf{L}$ is learned from a small subset of the columns of the data matrix. Second, the row-space (RS) of $\mathbf{L}$ is obtained using a small subset of the rows of $\mathbf{D}$. Unlike conventional decomposition that uses the entire data, we only utilize a small data sketch, and solve two low-dimensional optimization problems in lieu of one high-dimensional matrix decomposition problem (2) resulting in significant running time speed-ups.

To the best of our knowledge, it is shown here for the first time that the sufficient number of randomly sampled columns/rows scales linearly with the rank r and the coherency parameter of $\mathbf{L}$ even with uniform random sampling. Also, in contrast to the existing randomized approaches [5]–[7], which use blind uniform random sampling, we propose a new methodology for efficient column/row sampling. When the columns/rows of $\mathbf{L}$ are not distributed uniformly in the CS/RS of $\mathbf{L}$, which prevails much of the real world data, the proposed sampling approach is shown to achieve significant savings in data usage compared to uniform random sampling-based methods that require remarkable portions of the data. The proposed sampling algorithms can be independently used for feature selection from high-dimensional data.

In the presented approach, once the CS is learned, each column is decomposed efficiently and independently using the proposed randomized vector decomposition method. Unlike most existing approaches, which are batch-based, this unique feature enables applicability to online settings. The presented vector decomposition method can be independently used in many applications as an efficient vector decomposition algorithm or for efficient linear decoding [8]–[10].

A. Notation and definitions

We use bold-face upper-case letters to denote matrices and bold-face lower-case letters to denote vectors. Given a matrix $\mathbf{L}$, $\|\mathbf{L}\|$ denotes its spectral norm, $\|\mathbf{L}\|_F$ its Frobenius norm, and $\|\mathbf{L}\|_\infty$ the infinity norm, which is equal to the maximum absolute value of its elements. In an N-dimensional space, $\mathbf{e}_i$ is the i^{th} vector of the standard basis (i.e., the i^{th} element of $\mathbf{e}_i$ is equal to one and all the other elements are equal to zero).

This material is based upon work supported by the National Science Foundation under NSF grant No. CCF-1320547 and NSF CAREER Award CCF-1552497.

The authors are with the Department of Electrical Engineering and Computer Science, University of Central Florida,Orlando, FL 32816 USA (e-mail: mostafa@knights.ucf.edu, george.atia@ucf.edu).

The notation $A = [\,]$ denotes an empty matrix and the matrix

$$\mathbf{A} = [\mathbf{A}_1 \; \mathbf{A}_2 \; ... \; \mathbf{A}_n]$$

is the column-wise concatenation of the matrices $\{\mathbf{A}_i\}_{i=1}^n$. Random sampling refers to sampling without replacement.

II. Background and Related Work

A. Exact LR plus sparse matrix decomposition

The incoherence of the CS and RS of $\mathbf{L}$ is an important requirement for the identifiability of the decompostion problem in (1) [1], [2]. For the LR matrix $\mathbf{L}$ with rank r and compact SVD $\mathbf{L} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ (where $\mathbf{U} \in \mathbb{R}^{N_1 \times r}$, $\mathbf{\Sigma} \in \mathbb{R}^{r \times r}$ and $\mathbf{V} \in \mathbb{R}^{N_2 \times r}$), the incoherence condition is typically defined through the requirements [1], [2]

$$\begin{aligned} &\max_i \|\mathbf{U}^T \mathbf{e}_i\|_2^2 \le \frac{\mu r}{N_1} \;,\; \max_i \|\mathbf{V}^T \mathbf{e}_i\|_2^2 \le \frac{\mu r}{N_2} \\ &\text{and } \|\mathbf{U}\mathbf{V}^T\|_\infty \le \sqrt{\frac{\mu r}{N_2 N_1}} \end{aligned} \tag{3}$$

for some parameter μ that bounds the projection of the standard basis $\{\mathbf{e}_i\}$ onto the CS and RS. Other useful measures for the coherency of subspaces are given in [11] as,

$$\gamma(\mathbf{U}) = \sqrt{N_1} \max_{i,j} |\mathbf{U}(i,j)|, \gamma(\mathbf{V}) = \sqrt{N_2} \max_{i,j} |\mathbf{V}(i,j)|, \tag{4}$$

where $\gamma(\mathbf{U})$ and $\gamma(\mathbf{V})$ bound the coherency of the CS and the RS, respectively. When some of the elements of the orthonormal basis of a subspace are too large, the subspace is coherent with the standard vectors. Actually, it is not hard to show that $\max(\gamma(\mathbf{V}), \gamma(\mathbf{U})) \le \sqrt{\mu}$.

The decomposition of a data matrix into its LR and sparse components was analyzed in [1], [2], and sufficient conditions for exact recovery using the convex minimization (2) were derived. In [1], the sparsity pattern of the sparse matrix is selected uniformly at random following the so-called Bernoulli model to ensure that the sparse matrix is not LR with overwhelming probability. In this model, which is also used in this paper, each element of the sparse matrix can be nonzero independently with a constant probability. Without loss of generality (w.l.o.g.), suppose that $N_2 \le N_1$. The following lemma states the main result of [1].

Lemma 1 (Adapted from [1])**.** *Suppose that the support set of* $\mathbf{S}$ *follows the Bernoulli model with parameter* ρ. *The convex program (2) with* $\lambda = \frac{1}{\sqrt{N_1}}$ *yields the exact decomposition with probability at least* $1 - c_1 {N_1}^{-10}$ *provided that*

$$r \le \rho_r N_2 \mu^{-1} \left(\log(N_1)\right)^{-2} \quad , \quad \rho \le \rho_s \tag{5}$$

where ρ_s, c_1 *and* ρ_r *are numerical constants.*

The optimization problem in (2) is convex and can be solved using standard techniques such as interior point methods [2]. Although these methods have fast convergence rates, their usage is limited to small-size problems due to the high complexity of computing a step direction. Similar to the iterative shrinking algorithms for ℓ_1-norm and nuclear norm minimization, a family of iterative algorithms for solving the optimization problem (2) were proposed in [3], [4]. However, they also require working with the entire data. For example, the algorithm in [4] requires computing the SVD of an $N_1 \times N_2$ matrix in every iteration.

B. Randomized approaches

Owing to their inherent low-dimensional structures, the robust principal component analysis (PCA) and matrix decomposition problems can be conceivably solved using small data sketches, i.e., a small set of random observations of the data [6], [7], [12]–[15]. In [12], it was shown based on a simple degrees-of-freedom analysis that the LR and the sparse components can be precisely recovered using a small set of random linear measurements of $\mathbf{D}$. A convex program was proposed in [12] to recover these components using random matrix embedding with a polylogarithmic penalty factor in sample complexity, albeit the formulation also requires solving a high-dimensional optimization problem.

The iterative algorithms which solve (2) have complexity $\mathcal{O}(N_1 N_2 r)$ per iteration since they compute the partial SVD decomposition of $N_1 \times N_2$ dimensional matrices [4]. To reduce complexity, GoDec [16] uses a randomized method to efficiently compute the SVD, and the decomposition algorithm in [17] minimizes the rank of $\mathbf{\Phi L}$ instead of $\mathbf{L}$, where $\mathbf{\Phi}$ is a random projection matrix. However, these approaches do not have provable performance guarantees and their memory requirements scale with the full data dimensions. Another limitation of the algorithm in [17] is its instability since different random projections may yield different results.

The divide-and-conquer approach in [5] (and a similar algorithm in [18]), can achieve super-linear speedups over full-scale matrix decomposition. This approach forms an estimate of $\mathbf{L}$ by combining two low-rank approximations obtained from submatrices formed from sampled rows and columns of $\mathbf{D}$ using the generalized Nyström method [19]. Our approach also achieves super-linear speedups in decomposition, yet is fundamentally different from [5] and offers several advantages for the following reasons. First, our approach is a *subspace-pursuit approach* that focuses on subspace learning in a structure-preserving data sketch. Once the CS is learned, each column of the data is decomposed independently using a proposed randomized vector decomposition algorithm. Second, unlike [5], which is a batch approach that requires to store the entire data, the structure of the proposed approach naturally lends itself to online implementation (c.f. Section IV-E), which could be very beneficial for settings where the data comes in on the fly. Third, while the analysis provided in [5] requires roughly $\mathcal{O}(r^2 \mu^2 \max(N_1, N_2))$ random observations to ensure exact decomposition with high probability (whp), we show that the order of sufficient number of random observations depends linearly on the rank and the coherency parameter even if uniform random sampling is used. Fourth, the structure of the proposed approach enables us to leverage efficient sampling strategies for challenging and realistic scenarios in which the columns and rows of $\mathbf{L}$ are not uniformly distributed in their respective subspaces, or when the data exhibits additional structures (e.g. clustering structures) (c.f. Sections IV-B,IV-C). In such settings, the uniform random sampling used in [5] requires significantly larger amounts of data to carry out the decomposition.

III. Structure of the Proposed Approach and Theoretical Result

In this section, the structure of the proposed randomized decomposition method is presented. A step-by-step analysis of the proposed approach is provided and sufficient conditions for exact decomposition are derived. Theorem 5 stating the main theoretical result of the paper is presented at the end of this section. The proofs of the lemmas and the theorem are deferred to the appendix. Let us rewrite (1) as

$$\mathbf{D} = \mathbf{U}\mathbf{Q} + \mathbf{S}, \tag{6}$$

where $\mathbf{Q} = \mathbf{\Sigma}\mathbf{V}$. The representation matrix $\mathbf{Q} \in \mathbb{R}^{r \times N_2}$ is a full row rank matrix that contains the expansion of the columns of $\mathbf{L}$ in the orthonormal basis $\mathbf{U}$. The first step of the proposed approach aims to learn the CS of $\mathbf{L}$ using a subset of the columns of $\mathbf{D}$, and in the second step the representation matrix is obtained using a subset of the rows of $\mathbf{D}$.

Let $\mathcal{U}$ denote the CS of $\mathbf{L}$. Fundamentally, $\mathcal{U}$ can be obtained from a small subset of the columns of $\mathbf{L}$. However, since we do not have direct access to the LR matrix, a random subset of the columns of $\mathbf{D}$ is first selected. Hence, the matrix of sampled columns $\mathbf{D}_{s1}$ can be written as $\mathbf{D}_{s1} = \mathbf{D}\mathbf{S}_1$, where $\mathbf{S}_1 \in \mathbb{R}^{N_2 \times m_1}$ is the column sampling matrix and m_1 is the number of selected columns. The matrix of selected columns can be written as

$$\mathbf{D}_{s1} = \mathbf{L}_{s1} + \mathbf{S}_{s1}, \tag{7}$$

where $\mathbf{L}_{s1}$ and $\mathbf{S}_{s1}$ are its LR and sparse components, respectively. The idea is to decompose the sketch $\mathbf{D}_{s1}$ into its LR and sparse components to learn the CS of $\mathbf{L}$ from the CS of $\mathbf{L}_{s1}$. Note that the columns of $\mathbf{L}_{s1}$ are a subset of the columns of $\mathbf{L}$ since $\mathbf{L}_{s1} = \mathbf{L}\mathbf{S}_1$. Should we be able to decompose $\mathbf{D}_{s1}$ into its exact LR and sparse components (c.f. Lemma 3), we also need to ensure that the columns of $\mathbf{L}_{s1}$ span $\mathcal{U}$. The following lemma establishes that a small subset of the columns of $\mathbf{D}$ sampled uniformly at random contains sufficient information (i.e., the columns of the LR component of the sampled data span $\mathcal{U}$) if the RS is incoherent.

Lemma 2. *Suppose m_1 columns are sampled uniformly at random from the matrix $\mathbf{L}$ with rank r. If*

$$m_1 \geq r\gamma^2(\mathbf{V}) \max\left(c_2 \log r, c_3 \log\left(\frac{3}{\delta}\right)\right), \tag{8}$$

then the selected columns of the matrix $\mathbf{L}$ span the columns subspace of $\mathbf{L}$ with probability at least $(1-\delta)$ where c_2 and c_3 are numerical constants.

Thus, if $\gamma(\mathbf{V})$ is small (i.e., the RS is not coherent), a small set of randomly sampled columns can span $\mathcal{U}$. According to Lemma 2, if m_1 satisfies (8), then $\mathbf{L}$ and $\mathbf{L}_{s1}$ have the same CS whp. The following optimization problem (of dimensionality $N_1 m_1$) is solved to decompose $\mathbf{D}_{s1}$ into its LR and sparse components.

$$\begin{aligned} \min_{\dot{\mathbf{L}}_{s1}, \dot{\mathbf{S}}_{s1}} \quad & \frac{1}{\sqrt{N_1}} \|\dot{\mathbf{S}}_{s1}\|_1 + \|\dot{\mathbf{L}}_{s1}\|_* \\ \text{subject to} \quad & \dot{\mathbf{L}}_{s1} + \dot{\mathbf{S}}_{s1} = \mathbf{D}_{s1}. \end{aligned} \tag{9}$$

Thus, the columns subspace of the LR matrix can be recovered by finding the columns subspace of $\mathbf{L}_{s1}$. Our next lemma establishes that (9) yields the exact decomposition using roughly $m_1 = \mathcal{O}(\mu r)$ randomly sampled columns. To simplify the analysis, in the following lemma it is assumed that the CS of the LR matrix is sampled from the random orthogonal model [20], i.e., the columns of $\mathbf{U}$ are selected uniformly at random among all families of r-orthonormal vectors.

Lemma 3. *Suppose the columns subspace of $\mathbf{L}$ is sampled from the random orthogonal model, $\mathbf{L}_{s1}$ has the same column subspace of $\mathbf{L}$ and the support set of $\mathbf{S}$ follows the Bernoulli model with parameter ρ. In addition, assume that the columns of $\mathbf{D}_{s1}$ were sampled uniformly at random. If*

$$m_1 \geq \frac{r}{\rho_r} \mu' (\log N_1)^2 \quad \text{and} \quad \rho \leq \rho_s \,, \tag{10}$$

then (9) yields the exact decomposition with probability at least $1 - c_8 N_1^{-3}$, where

$$\mu' = \max\left(\frac{c_7 \max(r, \log N_1)}{r}, 6\gamma^2(\mathbf{V}), (c_9 \gamma(\mathbf{V}) \log N_1)^2\right) \tag{11}$$

and c_7, c_8 and c_9 are constant numbers provided that N_1 is greater than the RHS of first inequality of (10).

Therefore, according to Lemma 3 and Lemma 2, the CS of $\mathbf{L}$ can be obtained using roughly $O(r\mu)$ uniformly sampled data columns. Note that $m_1 \ll N_1$ for high-dimensional data as m_1 scales linearly with r. Hence, the requirement that N_1 is also greater than the RHS of first inequality of (10) is by no means restrictive and is naturally satisfied.

Suppose that (9) decomposes $\mathbf{D}_{s1}$ into its exact components and assume that $\mathcal{U}$ has been correctly identified. W.l.o.g., we can use $\mathbf{U}$ as an orthonormal basis for the learned CS. An arbitrary column $\mathbf{d}_i$ of $\mathbf{D}$ can be written as $\mathbf{d}_i = \mathbf{U}\mathbf{q}_i + \mathbf{s}_i$, where $\mathbf{q}_i$ and $\mathbf{s}_i$ are the corresponding columns of $\mathbf{Q}$ and $\mathbf{S}$, respectively. Thus, $\mathbf{d}_i - \mathbf{U}\mathbf{q}_i$ is a sparse vector. This suggests that $\mathbf{q}_i$ can be learned using the minimization

$$\min_{\hat{\mathbf{q}}_i} \|\mathbf{d}_i - \mathbf{U}\hat{\mathbf{q}}_i\|_1 \,, \tag{12}$$

where the ℓ_1-norm is used as a surrogate for the ℓ_0-norm to promote a sparse solution [8], [11]. The optimization problem (12) is similar to a system of linear equations with r unknown variables and N_1 equations. Since $r \ll N_1$, the idea is to learn $\mathbf{q}_i$ using only a small subset of the equations. Thus, we propose the following vector decomposition program

$$\min_{\hat{\mathbf{q}}_i} \|\mathbf{S}_2^T \mathbf{d}_i - \mathbf{S}_2^T \mathbf{U}\hat{\mathbf{q}}_i\|_1 \,, \tag{13}$$

where $\mathbf{S}_2 \in \mathbb{R}^{N_1 \times m_2}$ selects m_2 rows of $\mathbf{U}$ (and the corresponding m_2 elements of $\mathbf{d}_i$).

First, we have to ensure that the rank of $\mathbf{S}_2^T \mathbf{U}$ is equal to the rank of $\mathbf{U}$, for if $\mathbf{q}^*$ is the optimal point of (13), then $\mathbf{U}\mathbf{q}^*$ will be the LR component of $\mathbf{d}_i$. According to Lemma 2, $m_2 = \mathcal{O}(r\gamma(\mathbf{U}))$, is sufficient to preserve the rank of $\mathbf{U}$ when the rows are sampled uniformly at random. In addition, the following lemma establishes that if the rank of $\mathbf{U}$ is equal to the rank of $\mathbf{S}_2^T \mathbf{U}$, then the sufficient value of m_2 for (13) to yield the correct columns of $\mathbf{Q}$ whp is linear in r.

Lemma 4. *Suppose that the rank of $\mathbf{S}_2^T\mathbf{U}$ is equal to the rank of $\mathbf{L}$ and assume that the CS of $\mathbf{L}$ is sampled from the random orthogonal model. The optimal point of (13) is equal to $\mathbf{q}_i$ with probability at least $(1-3\delta)$ provided that*

$$\begin{aligned} &\rho \leq \frac{0.5}{r\beta\left(c_6\kappa\log\frac{N_1}{\delta}+1\right)} \ , \\ &m_2 \geq \max\left(\frac{2r\beta(\beta-2)\log\left(\frac{1}{\delta}\right)}{3(\beta-1)^2}\left(c_6\kappa\log\frac{N_1}{\delta}+1\right),\right. \\ &\qquad\qquad \left. c_5(\log\frac{N_1}{\delta})^2, \sqrt[6]{\frac{3}{\delta}}\right) \end{aligned} \tag{14}$$

where $\kappa = \frac{\log N_1}{r}$, c_5 and c_6 are constant numbers and β can be any real number greater than one.

Therefore, we can obtain the LR component of each column using a random subset of its elements. Since (12) is an ℓ_1-norm minimization, we can write the representation matrix learning problem as

$$\min_{\hat{\mathbf{Q}}} \|\mathbf{S}_2^T\mathbf{D} - \mathbf{S}_2^T\mathbf{U}\hat{\mathbf{Q}}\|_1. \tag{15}$$

Thus, (15) learns $\mathbf{Q}$ using a subset of the rows of $\mathbf{D}$ as $\mathbf{S}_2^T\mathbf{D}$ is the matrix formed from m_2 sampled rows of $\mathbf{D}$.

As such, we solve two low-dimensional subspace pursuit problems (9) and (15) of dimensions N_1m_1 and N_2m_2, respectively, instead of an N_1N_2-dimensional decomposition problem (2), and use a small random subset of the data to learn $\mathbf{U}$ and $\mathbf{Q}$. The table of Algorithm 1 explains the structure of the proposed approach.

We can readily state the following theorem which establishes sufficient conditions for Algorithm 1 to yield exact decomposition. In this theorem, it is assumed that the columns and rows are sampled uniformly at random. In the next section, an efficient method for column and row sampling is presented. In addition, a scheme for online implementation is proposed.

Theorem 5. *Suppose the columns subspace of the LR matrix is sampled from the random orthogonal model and the support set of $\mathbf{S}$ follows the Bernoulli model with parameter ρ. In addition, it is assumed that Algorithm 1 samples the columns and rows uniformly at random. If for any small $\delta > 0$,*

$$\begin{aligned} &m_1 \geq \max\left(r\gamma^2(\mathbf{V})\max\left(c_2\log r, c_3\log\frac{3}{\delta}\right),\right. \\ &\qquad\qquad \left.\frac{r}{\rho_r}\mu^{'}(\log N_1)^2\right) \\ &m_2 \geq \max\left(r\log N_1 \max\left(c_2^{'}\log r, c_3^{'}\log\frac{3}{\delta}\right),\right. \\ &\qquad \frac{2r\beta(\beta-2)\log\left(\frac{N_2}{\delta}\right)}{3(\beta-1)^2}\left(c_6\kappa\log\frac{N_1N_2}{\delta}+1\right), \\ &\qquad\qquad \left. c_5(\log\frac{N_1N_2}{\delta})^2, \sqrt[6]{\frac{3}{\delta}}\right) \\ &\rho \leq \min\left(\rho_s, \frac{0.5}{r\beta\left(c_6\kappa\log\frac{N_1N_2}{\delta}+1\right)}\right) \end{aligned} \tag{16}$$

where

$$\mu^{'} = \max\left(\frac{c_7\max(r,\log N_1)}{r}, 6\gamma^2(\mathbf{V}), (c_9\gamma(\mathbf{V})\log N_1)^2\right),$$
$$\kappa = \frac{\log N_1}{r},$$

$\{c_i\}_{i=1}^9$, $c_2^{'}$ and $c_3^{'}$ are constant numbers and β is any real number greater that one, then the proposed approach (Algorithm 1) yields the exact decomposition with probability at least $(1-5\delta-3rN_1^{-7})$ provided that N_1 is greater than the RHS of first inequality of (10).

Theorem (5) guarantees that the LR component can be obtained using a small subset of the data. The randomized approach has two main advantages. First, it significantly reduces the memory/storage requirements since it only uses a small data sketch and solves two low-dimensional optimization problems versus one large problem. Second, the proposed approach has $\mathcal{O}(\max(N_1,N_2)\times\max(m_1,m_2)\times r)$ per-iteration running time complexity, which is significantly lower than $\mathcal{O}(N_1N_2r)$ per iteration for full scale decomposition (2) [3], [4] implying remarkable speedups for big data. For instance, consider $\mathbf{U}$ and $\mathbf{Q}$ sampled from $\mathcal{N}(0,1)$, $r=5$, and $\mathbf{S}$ following the Bernoulli model with $\rho = 0.02$. For values of $N_1 = N_2$ equal to 500, 1000, 5000, 10^4 and 2×10^4, if $m_1 = m_2 = 10r$, the proposed approach yields the correct decomposition with 90, 300, 680, 1520 and 4800 - fold speedup, respectively, over directly solving (2).

Algorithm 1 Structure of Proposed Approach

Input: Data matrix $\mathbf{D} \in \mathbb{R}^{N_1\times N_2}$

1. Initialization: Form column sampling matrix $\mathbf{S}_1 \in \mathbb{R}^{N_2\times m_1}$ and row sampling matrix $\mathbf{S}_2 \in \mathbb{R}^{N_1\times m_2}$.

2. CS Learning

2.1 Column sampling: Matrix $\mathbf{S}_1$ samples m_1 columns of the given data matrix, $\mathbf{D}_{s1} = \mathbf{D}\mathbf{S}_1$.

2.2 CS learning: The convex program (9) is applied to the sampled columns $\mathbf{D}_{s1}$.

2.3 CS calculation: The CS is found as the columns subspace of the calculated LR component.

3. Representation Matrix Learning

3.1 Row sampling: Matrix $\mathbf{S}_2$ samples m_2 rows of the given data matrix, $\mathbf{D}_{s2} = \mathbf{S}_2^T\mathbf{D}$.

3.2 Representation matrix learning: The convex problem (15) is applied to the sampled rows to find the representation matrix.

Output: If $\hat{\mathbf{U}}$ is an orthonormal basis for the learned CS and $\hat{\mathbf{Q}}$ is the obtained representation matrix, then $\hat{\mathbf{L}} = \hat{\mathbf{U}}\hat{\mathbf{Q}}$ is the obtained LR component.

IV. Efficient Column/Row Sampling

In sharp contrast to randomized algorithms for matrix approximations rooted in numerical linear algebra (NLA) [21], [22], which seek to compute matrix approximations from sampled data using importance sampling, in matrix decomposition and robust PCA we do not have direct access to the LR matrix to measure how informative particular columns/rows are. As such, the existing randomized algorithms for matrix decomposition and robust PCA [5]–[7], [13] have predominantly relied upon uniform random sampling of columns/rows.

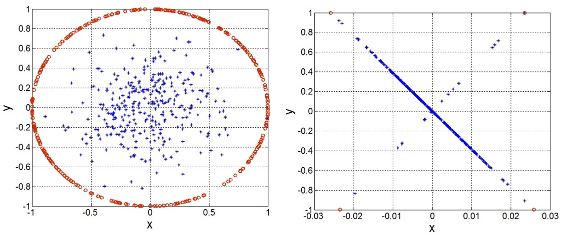

Fig. 1. Data distributions in a two-dimensional subspace. The red points are the normalized data points.

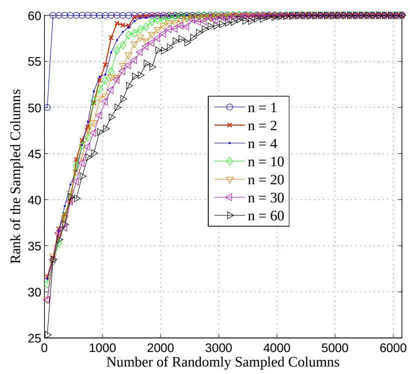

Fig. 2. The rank of a set of uniformly random sampled columns for different number of clusters.

In Section IV-A, we briefly describe the implications of non-uniform data distribution and show that uniform random sampling may not be favorable for data matrices exhibiting some structures that prevail much of the real datasets. In Section IV-B, we demonstrate an efficient column sampling strategy which will be integrated with the proposed decomposition method. The decomposition method with efficient column/row sampling is presented in Sections IV-C and IV-D.

A. Non-uniform data distribution

When data points lie in a low-dimensional subspace, a small subset of the points can span the subspace. However, uniform random sampling is only effective when the data points are distributed uniformly in the subspace. To clarify, Fig. 1 shows two scenarios for a set of data points in a two-dimensional subspace. In the left plot, the data points are distributed uniformly at random. In this case, two randomly sampled data points can span the subspace whp. In the right plot, 95 percent of the data lie on a one-dimensional subspace, thus we may not be able to capture the two-dimensional subspace from a small random subset of the data points.

In practice, the data points in a low-dimensional subspace may not be uniformly distributed, but rather exhibit some additional structures. A prevailing structure in many modern applications is clustered data [23], [24]. For example, user ratings for certain products (e.g. movies) in recommender systems are not only LR due to their inherent correlations, but also exhibit additional clustering structures owing to the similarity of the preferences of individuals from similar backgrounds (e.g. education, culture, or gender) [23], [25].

To further show that uniform random sampling falls short when the data points are not distributed uniformly in the subspace, consider a matrix $\mathbf{G} \in \mathbb{R}^{2000 \times 6150}$ generated as $\mathbf{G} = [\mathbf{G}_1 \; \mathbf{G}_2 \; ... \; \mathbf{G}_n]$. For $1 \leq i \leq \frac{n}{2}$, $\mathbf{G}_i = \mathbf{U}_i \mathbf{Q}_i$, where $\mathbf{U}_i \in \mathbb{R}^{2000 \times \frac{r}{n}}$, $\mathbf{Q}_i \in \mathbb{R}^{\frac{r}{n} \times \frac{200r}{n}}$. For $n/2 + 1 \leq i \leq n$, $\mathbf{G}_i = \mathbf{U}_i \mathbf{Q}_i$, where $\mathbf{U}_i \in \mathbb{R}^{2000 \times \frac{r}{n}}$, $\mathbf{Q}_i \in \mathbb{R}^{\frac{r}{n} \times \frac{5r}{n}}$. The elements of $\mathbf{U}_i$ and $\mathbf{Q}_i$ are sampled independently from a normal $\mathcal{N}(0, 1)$ distribution. The parameter r is set equal to 60, thus, the rank of $\mathbf{G}$ is equal to 60 whp. Fig. 2 illustrates the rank of the randomly sampled columns versus the number of sampled columns for different number of clusters n. As n increases, so does the required number of uniformly sampled columns. When $n = 60$, it turns out that we need to sample more than half of the columns to span the CS. As such, we cannot evade high-dimensionality with uniform random column/row sampling. In [23], it was shown that the RS coherency increases when the columns follow a more clustered distribution, and vice-versa. These observations match our theoretical result in Lemma 2, which established that the sufficient number of randomly sampled columns depends linearly on the coherency parameter of the RS.

B. Efficient column sampling method

Column sampling is widely used for dimensionality reduction and feature selection [15], [26], [27]. In the column sampling problem, the LR matrix (or the matrix whose span is to be approximated with a small set of its columns) is available. Thus, the columns are sampled based on their importance, measured by the so-called leverage scores [21], as opposed to blind uniform sampling. We refer the reader to [15], [27] and references therein for more information about efficient column sampling methods.

Next, we present a sampling approach which will be used in Section IV-C where the proposed decomposition algorithm with efficient sampling is presented. The proposed sampling strategy is inspired by the approach in [27] in the context of volume sampling. The table of Algorithm 2 details the presented column sampling procedure. Given a matrix $\mathbf{A}$ with rank r_A, the algorithm aims to sample a small subset of the columns of $\mathbf{A}$ that span its CS. The first column is sampled uniformly at random or based on a judiciously chosen probability distribution [21]. The next columns are selected sequentially so as to maximize the novelty to the span of the selected columns. As shown in step 2.2 of Algorithm 2, a design threshold τ is used to decide whether a given column brings sufficient novelty to the sampled columns by thresholding the ℓ_2-norm of its projection on the complement of the span of the sampled columns. The threshold τ is naturally set to zero in a noise-free setting. Once the selected columns are believed to span the CS of $\mathbf{A}$, they are removed from $\mathbf{A}$. This procedure is repeated C times (using the remaining columns). In each time, the algorithm finds r columns which span the CS of $\mathbf{A}$. After every iteration, the rank of the matrix of remaining columns is bounded above by r_A. As such, the algorithm samples approximately $m_1 \approx C r_A$ columns in total. In the proposed decomposition method with efficient column/row sampling (presented in Section IV-C), we set C large enough to ensure that the selected columns form a low rank matrix.

Algorithm 2 Efficient Sampling from LR Matrices

Input: Matrix $\mathbf{A}$.

1. Initialize

1.1 The parameter C is chosen as an integer greater than or equal to one. The algorithm finds C sets of linearly dependent columns.

1.2 Set $\mathcal{I} = \emptyset$ as the index set of the sampled columns and set $v = \tau$, $\mathbf{B} = \mathbf{A}$ and $\mathbf{C} = [\,]$.

2. Repeat C Times

2.1 Let $\mathbf{b}$ be a non-zero randomly sampled column from $\mathbf{B}$ with index i_b. Update $\mathbf{C}$ and $\mathcal{I}$ as $\mathbf{C} = [\mathbf{C} \;\; \mathbf{b}]$, $\mathcal{I} = \{\mathcal{I}\,,\, i_b\}$.

2.2 While $v \geq \tau$

2.2.1 Set $\mathbf{E} = \mathbf{P}_c \mathbf{B}$, where $\mathbf{P}_c$ is the projection matrix onto the complement space of span$(\mathbf{C})$.

2.2.2 Define $\mathbf{f}$ as the column of $\mathbf{E}$ with the maximum ℓ_2-norm with index i_f. Update $\mathbf{C}$, $\mathcal{I}$ and v as $\mathbf{C} = [\mathbf{C} \;\; \mathbf{f}]$, $\mathcal{I} = \{\mathcal{I}\,,\, i_f\}$ and $v = \|\mathbf{f}\|_2$.

2.2 End While

2.3 Set $\mathbf{C} = [\,]$ and set $\mathbf{B}$ equal to $\mathbf{A}$ with the columns indexed by $\mathcal{I}$ set to zero.

2. End Repeat

Output: The set $\mathcal{I}$ contains the indices of the selected columns.

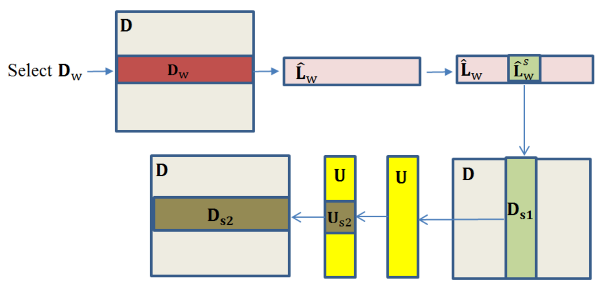

Fig. 3. Visualization of the matrices defined in Section IV-C. Matrix $\mathbf{D}_w$ is selected randomly or using Algorithm 3 described in Section IV-D.

C. Proposed decomposition algorithm with efficient sampling

In this section, we develop a modified decomposition algorithm that replaces uniform random sampling with the efficient column/row sampling method (Algorithm 2). In Section V, it is shown that the proposed technique can remarkably reduce the sampling requirement. We consider a setting wherein the data points (the columns of $\mathbf{L}$) are not uniformly distributed, rather they admit an additional structure (such as clustering), wherefore a small subset of uniformly sampled columns is not likely to span the CS. However, we assume that the *rows* of $\mathbf{L}$ are distributed well enough, in the sense that they do not much align along any specific directions, such that $C_r r$ rows of $\mathbf{L}$ sampled uniformly at random span its RS whp, for some constant C_r. In Section IV-D, we dispense with this assumption. The proposed decomposition algorithm rests on three key ideas detailed next.

1) Informative column sampling: The first important idea underlying the proposed sampling approach is to start sampling along the dimension that has the better distribution. For instance, in the example considered in Section IV-A, the columns of $\mathbf{G}$ admit a clustering structure. However, the CS of $\mathbf{G}$ is a random r-dimensional subspace, which means that the rows of $\mathbf{G}$ are distributed uniformly at random in the RS of $\mathbf{G}$. Thus, in this case we start with row sampling. The main intuition is that while almost 60 randomly sampled rows of $\mathbf{G}$ span the RS, a considerable portion of the columns (almost 4000) should be sampled to capture the CS as shown in Fig. 2. As another example, consider an extreme scenario where only two columns of $\mathbf{G} \in \mathbb{R}^{1000 \times 1000}$ are non-zero. In this case, with random sampling we need to sample almost all the columns to ensure that the sampled columns span the CS of $\mathbf{G}$. But, if the non-zero columns are non-sparse, a small subset of randomly chosen rows of $\mathbf{G}$ will span its row space.

Let $\hat{r}$ denote a known upper bound on r. Such knowledge is often available as side information depending on the particular application. For instance, facial images under varying illumination and facial expressions are known to lie on a special low-dimensional subspace [28]. For visualization, Fig. 3 provides a simplified illustration of the matrices defined in this section. We sample $C_r \hat{r}$ rows of $\mathbf{D}$ uniformly at random. Let $\mathbf{D}_w \in \mathbb{R}^{(C_r \hat{r}) \times N_2}$ denote the matrix of sampled rows. We choose C_r sufficiently large to ensure that the non-sparse component of $\mathbf{D}_w$ is a LR matrix. Define $\mathbf{L}_w$, assumably with rank r, as the LR component of $\mathbf{D}_w$. If we locate a subset of the columns of $\mathbf{L}_w$ that span its CS, the corresponding columns of $\mathbf{L}$ would span its CS. To this end, the convex program (2) is applied to $\mathbf{D}_w$ to extract its LR component denoted $\hat{\mathbf{L}}_w$. Then, Algorithm 2 is applied to $\hat{\mathbf{L}}_w$ to find a set of informative columns by sampling $m_1 \approx C\hat{r}$ columns. In Remark 1, we discuss how to choose C in the algorithm. Define $\hat{\mathbf{L}}_w^s$ as the matrix of columns selected from $\hat{\mathbf{L}}_w$. The matrix $\mathbf{D}_{s1}$ is formed using the columns of $\mathbf{D}$ corresponding to the sampled columns of $\hat{\mathbf{L}}_w$.

2) CS learning: Similar to the CS learning step of Algorithm 1, we can obtain the CS of $\mathbf{L}$ by decomposing $\mathbf{D}_{s1}$. However, we propose to leverage valuable information in the matrix $\hat{\mathbf{L}}_w^s$ in decomposing $\mathbf{D}_{s1}$. In particular, if $\mathbf{D}_w$ is decomposed correctly, the RS of $\hat{\mathbf{L}}_w^s$ would be same as that of $\mathbf{L}_{s1}$ given that the rank of $\mathbf{L}_w$ is equal to r. Let $\mathbf{V}_{s1}$ be an orthonormal basis for the RS of $\hat{\mathbf{L}}_w^s$. Thus, in order to learn the CS of $\mathbf{D}_{s1}$, we only need to solve

$$\min_{\hat{\mathbf{U}}} \|\mathbf{D}_{s1} - \hat{\mathbf{U}}\mathbf{V}_{s1}^T\|_1 \,. \tag{17}$$

Remark 1. *Define $\mathbf{d}_{s1}^i$ as the i^{th} row of $\mathbf{D}_{s1}$. According to (17), $\mathbf{U}^i$ (the i^{th} row of $\mathbf{U}$) is obtained as the optimal point of*

$$\min_{\hat{\mathbf{U}}^i} \|(\mathbf{d}_{s1}^i)^T - \mathbf{V}_{s1}(\hat{\mathbf{U}}^i)^T\|_1 \,. \tag{18}$$

Based on the analysis provided for the proof of Lemma 4, the optimal point of (18) is equal to $\mathbf{U}^i$ if $m_1 \geq r + \eta \|\mathbf{S}_{s1}^i\|_0$, where $\mathbf{S}_{s1}^i$ is the i^{th} row of $\mathbf{S}_{s1}$, $\|\mathbf{S}_{s1}^i\|_0$ the number of non-zero elements of $\mathbf{S}_{s1}^i$, and η a real number which depends on the coherency of the subspace spanned by $\mathbf{V}_{s1}$. Thus, here C is determined based on the rank of $\mathbf{L}$ and the sparsity of $\mathbf{S}$, i.e., $Cr - r$ has to be sufficiently greater than the expected value for the number of non-zero elements of the rows of $\mathbf{S}_{s1}$.

Remark 2. *We note that the convex algorithm (2) may not always yield accurate decomposition of $\mathbf{D}_w$ since structured data may not be sufficiently incoherent [2], [23] suggesting that the decomposition step can be further improved. Let $\mathbf{D}_w^s$*

be the matrix consisting of the columns of $\mathbf{D}_w$ corresponding to the columns selected from $\hat{\mathbf{L}}_w$ to form $\hat{\mathbf{L}}_w^s$. According to our investigations, an improved $\mathbf{V}_{s1}$ can be obtained by applying the decomposition algorithm presented in [29] to $\mathbf{D}_w^s$ then use the RS of $\hat{\mathbf{L}}_w^s$ as an initial guess for the RS of the non-sparse component of $\mathbf{D}_w^s$. Since $\mathbf{D}_w^s$ is low-dimensional (roughly $\mathcal{O}(r) \times \mathcal{O}(r)$ dimensional matrix), this extra step is a low complexity operation.

3) Representation matrix learning: Suppose that the CS of $\mathbf{L}$ was learned correctly, i.e., the span of the optimal point of (17) is equal to the span of $\mathbf{U}$. Thus, we use $\mathbf{U}$ as a basis for the learned CS. Now we leverage the information embedded in $\mathbf{U}$ to select the informative rows. Algorithm 2 is applied to $\mathbf{U}^T$ to locate $m_2 \approx Cr$ rows of $\mathbf{U}$. Thus, we form the matrix $\mathbf{D}_{s2}$ from the rows of $\mathbf{D}$ corresponding to the selected rows of $\mathbf{U}$. Thus, the representation matrix is learned as

$$\min_{\hat{\mathbf{Q}}} \|\mathbf{D}_{s2} - \mathbf{U}_{s2}\hat{\mathbf{Q}}\|_1 , \tag{19}$$

where $\mathbf{U}_{s2} \in \mathbb{R}^{m_2 \times r}$ is the matrix of the selected rows of $\mathbf{U}$. Subsequently, the LR matrix can be obtained from the learned CS and the representation matrix.

D. Column/Row sampling from sparsely corrupted data

In Section IV-C, we assumed that the LR component of $\mathbf{D}_w$ has rank r. However, if the rows are not well-distributed, a reasonably sized random subset of the rows may not span the RS of $\mathbf{L}$. Here, we present a sampling approach which can find the informative columns/rows even when both the columns and the rows exhibit clustering structures such that a small random subset of the columns/rows of $\mathbf{L}$ cannot span its CS/RS. The algorithm presented in this section (Algorithm 3) can be independently used as an efficient sampling approach from big data. In this paper, we use Algorithm 3 to form $\mathbf{D}_w$ if both the columns and rows exhibit clustering structures.

The table of Algorithm 3, Fig. 4 and its caption provide the details of the proposed sampling approach and the definitions of the used matrices. We start the cycle from the position marked “I” in Fig. 4 with $\mathbf{D}_w$ formed according to the initialization step of Algorithm 3. For ease of exposition, assume that $\hat{\mathbf{L}}_w = \mathbf{L}_w$ and $\hat{\mathbf{L}}_c = \mathbf{L}_c$, i.e., $\mathbf{D}_w$ and $\mathbf{D}_c$ are decomposed correctly. The matrix $\hat{\mathbf{L}}_w^s$ is the informative columns of $\hat{\mathbf{L}}_w$. Thus, the rank of $\hat{\mathbf{L}}_w^s$ is equal to the rank of $\hat{\mathbf{L}}_w$. Since $\hat{\mathbf{L}}_w = \mathbf{L}_w$, $\hat{\mathbf{L}}_w^s$ is a subset of the rows of $\mathbf{L}_c$. If the rows of $\mathbf{L}$ exhibit a clustering structure, it is likely that rank$(\hat{\mathbf{L}}_w^s) <$ rank$(\mathbf{L}_c)$. Thus, rank$(\mathbf{L}_w) <$ rank$(\mathbf{L}_c)$. We continue one cycle of the algorithm by going through steps 1, 2 and 3 of Fig. 4 to update $\mathbf{D}_w$. Using a similar argument, we see that the rank of an updated $\mathbf{L}_w$ will be greater than the rank of $\mathbf{L}_c$. Thus, if we run more cycles of the algorithm – each time updating $\mathbf{D}_w$ and $\mathbf{D}_c$ – the rank of $\mathbf{L}_w$ and $\mathbf{L}_c$ will increase. As detailed in the table of Algorithm 3, we stop if the dimension of the span of the obtained LR component does not change in T consecutive iterations. While there is no guarantee that the rank of $\mathbf{L}_w$ will converge to r (it can converge to a value smaller than r), our investigations have shown that Algorithm 3 performs quite well and the RS of $\mathbf{L}_w$ converges to the RS of $\mathbf{L}$ in few steps. We have also found that adding some randomly sampled columns (rows) to $\mathbf{D}_c(\mathbf{D}_w)$ can effectively avert converging to a lower dimensional subspace. For instance, some randomly sampled columns can be added to $\mathbf{D}_c$, which was obtained by applying Algorithm 2 to $\hat{\mathbf{L}}_w$.

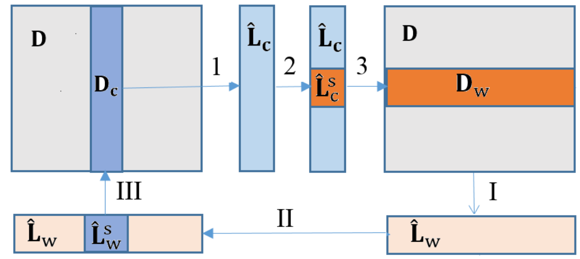

Fig. 4. Visualization of Algorithm 3. We run few cycles of the algorithm and stop when the rank of the LR component does not change over T consecutive steps. One cycle of the algorithm starts from the point marked “I” and proceeds as follows. **I**: Matrix $\mathbf{D}_w$ is decomposed and $\hat{\mathbf{L}}_w$ is the obtained LR component of $\mathbf{D}_w$. **II**: Algorithm 2 is applied to $\hat{\mathbf{L}}_w$ to select the informative columns of $\hat{\mathbf{L}}_w$. $\hat{\mathbf{L}}_w^s$ is the matrix of columns selected from $\hat{\mathbf{L}}_w$. **III**: Matrix $\mathbf{D}_c$ is formed from the columns of $\mathbf{D}$ that correspond to the columns of $\hat{\mathbf{L}}_w^s$. **1**: Matrix $\mathbf{D}_c$ is decomposed and $\hat{\mathbf{L}}_c$ is the obtained LR component of $\mathbf{D}_c$. **2**: Algorithm 2 is applied to $\hat{\mathbf{L}}_c^T$ to select the informative rows of $\hat{\mathbf{L}}_c$. $\hat{\mathbf{L}}_c^s$ is the matrix of rows selected from $\hat{\mathbf{L}}_c$. **3**: Matrix $\mathbf{D}_w$ is formed as the rows of $\mathbf{D}$ corresponding to the rows used to form $\hat{\mathbf{L}}_c^s$.

Algorithm 3 Efficient Column/Row Sampling from Sparsely Corrupted LR Matrices

1. Initialization

Form $\mathbf{D}_w \in \mathbb{R}^{C_r \hat{r} \times N_2}$ by randomly choosing $C_r \hat{r}$ rows of $\mathbf{D}$. Initialize $k = 1$ and set T equal to an integer greater than 1.

2. While $k > 0$

2.1 Sample the most informative columns

2.1.1 Obtain $\hat{\mathbf{L}}_w$ via (2) as the LR component of $\mathbf{D}_w$.

2.1.2 Apply Algorithm 2 to $\hat{\mathbf{L}}_w$ with $C = C_r$.

2.1.3 Form the matrix $\mathbf{D}_c$ from the columns of $\mathbf{D}$ corresponding to the sampled columns of $\hat{\mathbf{L}}_w$.

2.2 Sample the most informative rows

2.2.1 Obtain $\hat{\mathbf{L}}_c$ via (2) as the LR component of $\mathbf{D}_c$.

2.2.2 Apply Algorithm 2 to $\hat{\mathbf{L}}_c^T$ with $C = C_r$.

2.2.3 Form the matrix $\mathbf{D}_w$ from the rows of $\mathbf{D}$ corresponding to the sampled rows of $\hat{\mathbf{L}}_c$.

2.3 If the dimension of the RS of $\hat{\mathbf{L}}_w$ does not increase in T consecutive iterations, set $k = 0$ to stop the algorithm.

2. End While

Output: The matrices $\mathbf{D}_w$ and $\hat{\mathbf{L}}_w$ can be used for column sampling in the first step of the Algorithm presented in Section IV-C.

Algorithm 3 was found to converge in a very small number of iterations (typically less than 4 iterations). Thus, even when Algorithm 3 is used to form the matrix $\mathbf{D}_w$, the order of complexity of the proposed decomposition method with efficient column/row sampling (presented in Section IV-C) is roughly $\mathcal{O}(\max(N_1, N_2) r^2)$.

E. Online Implementation

The proposed decomposition approach consists of two main steps, namely, learning the CS of the LR component then

decomposing the columns independently. This structure lends itself to online implementation, which could be very beneficial in settings where the data arrives on the fly. The idea is to first learn the CS of the LR component from a small batch of the data and keep tracking the CS. Since the CS is being tracked, any new data column can be decomposed based on the updated subspace. The table of Algorithm 4 details the proposed online matrix decomposition algorithm, where $\mathbf{d}_t$ denotes the t^{th} received data column.

Algorithm 4 uses a parameter n_u which determines the rate at which the algorithm updates the CS of the LR component. For instance, if $n_u = 20$, then the CS is updated every 20 new data columns (step 2.2 of Algorithm 4). The parameter n_u has to be set in accordance with the rate of the change of the subspace of the LR component; a small value for n_u is used if the subspace is changing rapidly. The parameter n_s determines the number of columns last received that are used to update the CS. If the subspace changes rapidly, the older columns may be less relevant to the current subspace, hence a small value for n_s is used. On the other hand, when the data is noisy and the subspace changes at a slower rate, choosing a larger value for n_s can lead to more accurate estimation of the CS.

Algorithm 4 Online Implementation

1. Initialization

1.1 Set the parameters n_u and n_s equal to integers greater than or equal to one.

1.2 Form $\mathbf{D}_0 \in \mathbb{R}^{N_1 \times (C_r \hat{r})}$ as

$$\mathbf{D}_0 = [\mathbf{d}_1 \; \mathbf{d}_2 \; ... \; \mathbf{d}_{C_r \hat{r}}].$$

Decompose $\mathbf{D}_0$ using (2) and obtain the CS of its LR component. Define $\mathbf{U}_o$ as the learned CS, $\mathbf{Q}_o$ the appropriate representation matrix and $\hat{\mathbf{S}}$ the obtained sparse component of $\mathbf{D}_0$.

1.3 Apply Algorithm 2 to $\mathbf{U}_o^T$ to construct the row sampling matrix $\mathbf{S}_2$.

2. For any new data column $\mathbf{d}_t$ do

2.1 Decompose $\mathbf{d}_t$ as

$$\min_{\hat{\mathbf{q}}_t} \|\mathbf{S}_2^T \mathbf{d}_t - \mathbf{S}_2^T \mathbf{U}_o \hat{\mathbf{q}}_t\|_1 \,, \tag{20}$$

and update

$\mathbf{Q}_o \leftarrow [\mathbf{Q}_o \;\; \mathbf{q}_t^*]$, $\hat{\mathbf{S}} \leftarrow [\hat{\mathbf{S}} \;\; (\mathbf{d}_t - \mathbf{U}_o \mathbf{q}_t^*)]$, where $\mathbf{q}_t^*$ is the optimal point of (20).

2.2 If the remainder of $\frac{t}{n_u}$ is equal to zero, update $\mathbf{U}_o$ as

$$\min_{\hat{U}_o} \|\mathbf{D}_t - \hat{\mathbf{U}}_o \mathbf{Q}_o^t\|_1 \,, \tag{21}$$

where $\mathbf{Q}_o^t$ is the last $n_s \hat{r}$ columns of $\mathbf{Q}_o$ and $\mathbf{D}_t$ is the matrix formed from the last $n_s \hat{r}$ received data columns. Apply Algorithm 2 to the new $\mathbf{U}_o^T$ to update the row sampling matrix $\mathbf{S}_2$.

2. End For

Output The matrix $\hat{\mathbf{S}}$ as the obtained sparse matrix, $\hat{\mathbf{L}} = \mathbf{D} - \hat{\mathbf{S}}$ as the obtained LR matrix and $\mathbf{U}_o$ as the current basis for the CS of the LR component.

F. Noisy Data

In practice, noisy data can be modeled as

$$\mathbf{D} = \mathbf{L} + \mathbf{S} + \mathbf{N} \,, \tag{22}$$

where $\mathbf{N}$ is an additive noise component. In [30], it was shown that the program

$$\begin{aligned} \min_{\hat{\mathbf{L}}, \hat{\mathbf{S}}} \quad & \lambda \|\hat{\mathbf{S}}\|_1 + \|\hat{\mathbf{L}}\|_* \\ \text{subject to} \quad & \left\|\hat{\mathbf{L}} + \hat{\mathbf{S}} - \mathbf{D}\right\|_F \leq \epsilon_n \,, \end{aligned} \tag{23}$$

can recover the LR and sparse components with an error bound that is proportional to the noise level. The parameter ϵ_n has to be chosen based on the noise level. This modified version can be used in the proposed algorithms to account for the noise. Similarly, to account for the noise in the representation learning problem (15), the ℓ_1-norm minimization problem can be modified as follows:

$$\min_{\hat{\mathbf{Q}}, \hat{\mathbf{E}}} \|\mathbf{S}_2^T \mathbf{D} - \mathbf{S}_2^T \mathbf{U} \hat{\mathbf{Q}} - \hat{\mathbf{E}}\|_1 \quad \text{subject to} \quad \|\hat{\mathbf{E}}\|_F \leq \delta_n. \tag{24}$$

$\hat{\mathbf{E}} \in \mathbb{R}^{m_2 \times N_2}$ is used to cancel out the effect of the noise and the parameter δ_n is chosen based on the noise level [31].

V. Numerical Simulations

In this section, we present some numerical simulations to study the performance of the proposed randomized decomposition method. First, we present a set of simulations confirming our analysis which established that the sufficient number of sampled columns/rows is linear in r. Then, we compare the proposed approach to the state-of-the-art randomized algorithm [5] and demonstrate that the proposed sampling strategy can lead to notable improvement in performance. We then provide an illustrative example to showcase the effectiveness of our approach on real video frames for background subtraction and activity detection. Given the structure of the proposed approach, it is shown that side information can be leveraged to further simplify the decomposition task. In addition, a numerical example is provided to examine the performance of Algorithm 3. Finally, we investigate the performance of the online algorithm and show that the proposed online method can successfully track the underlying subspace.

In all simulations, the Augmented Lagrange multiplier (ALM) algorithm [1], [4] is used to solve the optimization problem (2). In addition, the ℓ_1-magic routine [32] is used to solve the ℓ_1-norm minimization problems. It is important to note that in all the provided simulations (except in Section V-D), the convex program (2) that operates on the entire data can yield correct decomposition with respect to the considered criteria. Thus, if the randomized methods cannot yield correct decomposition, it is because they fall short of acquiring the essential information through sampling.

A. Phase transition plots

In this section, we investigate the required number of randomly sampled columns/rows. The LR matrix is generated as a product $\mathbf{L} = \mathbf{U}_r \mathbf{Q}_r$, where $\mathbf{U}_r \in \mathbb{R}^{N_1 \times r}$ and $\mathbf{Q}_r \in \mathbb{R}^{r \times N_2}$. The elements of $\mathbf{U}_r$ and $\mathbf{Q}_r$ are sampled independently from a standard normal $\mathcal{N}(0,1)$ distribution. The sparse matrix $\mathbf{S}$ follows the Bernoulli model with $\rho = 0.02$. In this experiment, Algorithm 1 is used and the column/rows are sampled uniformly at random.

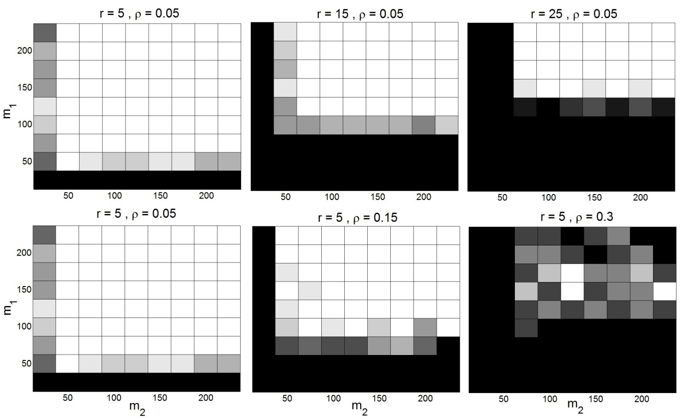

Fig. 5. Phase transition plots for various rank and sparsity levels. White designates successful decomposition and black designates incorrect decomposition.

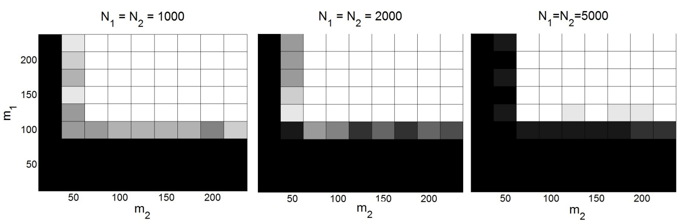

Fig. 6. Phase transition plots for various data matrix dimensions.

Fig. 5 shows the phase transition plots for different numbers of randomly sampled rows/columns. In this simulation, the data is a 1000×1000 matrix. For each (m_1, m_2), we generate 10 random realizations. A trial is considered successful if the recovered LR matrix $\hat{\mathbf{L}}$ satisfies $\frac{\|\mathbf{L}-\hat{\mathbf{L}}\|_F}{\|\mathbf{L}\|_F} \leq 5 \times 10^{-3}$. It is clear that the required number of sampled columns/rows increases as the rank or the sparsity parameter ρ are increased. When the sparsity parameter is increased to 0.3, the proposed algorithm can hardly yield correct decomposition. Actually, in this case the matrix $\mathbf{S}$ is no longer a sparse matrix.

The top row of Fig. 5 confirms that the sufficient values for m_1 and m_2 are roughly linear in r. For instance, when the rank is increased from 5 to 25, the required value for m_1 increases from 30 to 140. In this experiment, the column and RS of $\mathbf{L}$ are sampled from the random orthogonal model. Thus, the CS and RS have small coherency whp [20]. Therefore, the important factor governing the sample complexity is the rank of $\mathbf{L}$. Indeed, Fig. 6 shows the phase transition for different sizes of the data matrix when the rank of $\mathbf{L}$ is fixed. One can see that the required values for m_1 and m_2 are almost independent of the size of the data confirming our analysis.

B. Efficient column/row sampling

In this experiment, the algorithm presented in Section IV-C is compared to the randomized decomposition algorithm in [5]. It is shown that the proposed sampling strategy can effectively reduce the required number of sampled columns/rows, and makes the proposed method remarkably robust to structured data. In this experiment, $\mathbf{D}$ is a 2000×4200 matrix. The LR component is generated as

$$\mathbf{L} = [\mathbf{G}_1 \; \mathbf{G}_2 \; ... \; \mathbf{G}_n] \, .$$

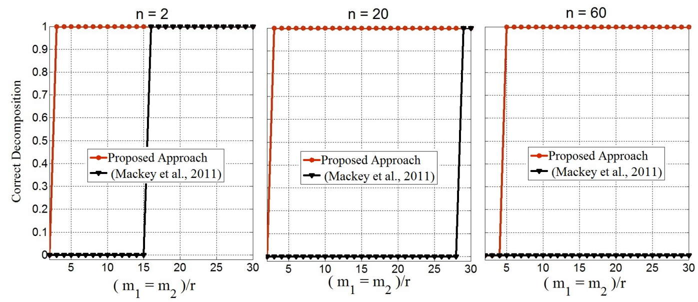

Fig. 7. Performance of the proposed approach and the randomized algorithm in [5]. A value 1 indicates correct decomposition and a value 0 indicates incorrect decomposition.

For $1 \leq i \leq \frac{n}{2}$,

$$\mathbf{G}_i = \mathbf{U}_i \mathbf{Q}_i \; ,$$

where $\mathbf{U}_i \in \mathbb{R}^{2000 \times \frac{r}{n}}$, $\mathbf{Q}_i \in \mathbb{R}^{\frac{r}{n} \times \frac{130r}{n}}$ and the elements of $\mathbf{U}_i$ and $\mathbf{Q}_i$ are sampled independently from a normal distribution $\mathcal{N}(0,1)$. For $n/2 + 1 \leq i \leq n$,

$$\mathbf{G}_i = 13 \mathbf{U}_i \mathbf{Q}_i \; ,$$

where $\mathbf{U}_i \in \mathbb{R}^{2000 \times \frac{r}{n}}$, $\mathbf{Q}_i \in \mathbb{R}^{\frac{r}{n} \times \frac{10r}{n}}$, and the elements of $\mathbf{U}_i$ and $\mathbf{Q}_i$ are sampled independently from an $\mathcal{N}(0,1)$ distribution. We set r equal to 60; thus, the rank of $\mathbf{L}$ is equal to 60 whp. The sparse matrix $\mathbf{S}$ follows the Bernoulli model and each element of $\mathbf{S}$ is non-zero with probability 0.02. In this simulation, we do not use Algorithm 3 to form $\mathbf{D}_w$. The matrix $\mathbf{D}_w$ is formed from 300 uniformly sampled *rows* of $\mathbf{D}$.

We evaluate the performance of the algorithm for different values of n, i.e., different number of clusters. Fig. 7 shows the performance of the proposed approach and the approach in [5] for different values of m_1 and m_2. For each value of $m_1 = m_2$, we compute the error in LR matrix recovery $\frac{\|\mathbf{L}-\hat{\mathbf{L}}\|_F}{\|\mathbf{L}\|_F}$ averaged over 10 independent runs, and conclude that the algorithm can yield correct decomposition if the average error is less than 0.01. In Fig. 7, the values 0, 1 designate incorrect and correct decomposition, respectively. It can be seen that the presented approach requires a significantly smaller number of samples to yield the correct decomposition. This is due to the fact that the randomized algorithm [5] samples both the columns and rows uniformly at random and independently. In sharp contrast, we use $\hat{\mathbf{L}}_w$ to find the most informative columns to form $\mathbf{D}_{s1}$, and also leverage the information embedded in the CS to find the informative rows to from $\mathbf{D}_{s2}$. One can see that when $n = 60$, [5] cannot yield correct decomposition even when $m_1 = m_2 = 1800$.

C. Vector decomposition for background subtraction

The LR plus sparse matrix decomposition can be effectively used to detect a moving object in a stationary background [1], [33]. The background is modeled as a LR matrix and the moving object as a sparse matrix. Since videos are typically high dimensional objects, standard algorithms can be quite slow for such applications. Our algorithm is a good candidate for such a problem as it reduces the dimensionality significantly. The decomposition problem can be further simplified by leveraging

Fig. 8. Stationary background.

Fig. 9. Two frames of a video taken in a lobby. The first column displays the original frames. The second and third columns display the LR and sparse components recovered using the proposed approach.

prior information about the stationary background. In particular, we know that the background does not change or we can construct it with some pre-known dictionary. For example, consider the video from [34], which was also used in [1]. Few frames of the stationary background are illustrated in Fig. 8. Thus, we can simply form the CS of the LR matrix using these frames which can describe the stationary background in different states. Accordingly, we just need to learn the representation matrix. As such, the background subtraction problem is simplified to a vector decomposition problem. Fig. 9 shows that the proposed method successfully separates the background and the moving objects. In this experiment, 500 randomly sampled rows are used (i.e., 500 randomly sampled pixels) for the representation matrix learning (15). While the running time of our approach is just few milliseconds, it takes almost half an hour if we use (2) to decompose the video file [1].

D. Alternating algorithm for column sampling

In this section, we investigate the performance of Algorithm 3 for column sampling. The rank of the selected columns is shown to converge to the rank of $\mathbf{L}$ even when both the rows and columns of $\mathbf{L}$ exhibit a highly structured distribution. To generate the LR matrix $\mathbf{L}$ we first generate a matrix $\mathbf{G}$ as in Section IV-A but setting $r = 100$. Then, we construct the matrix $\mathbf{U}_g$ from the first r right singular vectors of $\mathbf{G}$. We then generate $\mathbf{G}$ in a similar way and set $\mathbf{V}_g$ equal to the first r right singular vectors of $\mathbf{G}$. Let the matrix $\mathbf{L} = \mathbf{U}_g \mathbf{V}_g^T$. For example, for $n = 100$, $\mathbf{L} \in \mathbb{R}^{10250 \times 10250}$. Note that the resulting LR matrix is nearly sparse since in this simulation we consider a very challenging scenario in which both the columns and rows of $\mathbf{L}$ are highly structured and coherent. Thus, in this simulation we set the sparse matrix equal to zero and use Algorithm 3 as follows. The matrix $\mathbf{D}_c$ is formed using 300 columns sampled uniformly at random and the following steps are performed iteratively:

1. Apply Algorithm 2 to $\mathbf{D}_c^T$ with $C = 3$ to sample approximately $3r$ columns of $\mathbf{D}_c^T$ and form $\mathbf{D}_w$ from the rows of $\mathbf{D}$ corresponding to the selected rows of $\mathbf{D}_c$.
2. Apply Algorithm 2 to $\mathbf{D}_w$ with $C = 3$ to sample approximately $3r$ columns of $\mathbf{D}_w$ and form $\mathbf{D}_c$ from the columns of $\mathbf{D}$ corresponding to the selected columns of $\mathbf{D}_c$. Fig. 10 shows the rank of $\mathbf{D}_c$ after each iteration. It is evident that the algorithm converges to the rank of $\mathbf{L}$ in less than 3 iterations even for $n = 100$ clusters. For all values of n, i.e., $n \in \{2, 50, 60\}$, the data is a 10250×10250 matrix.

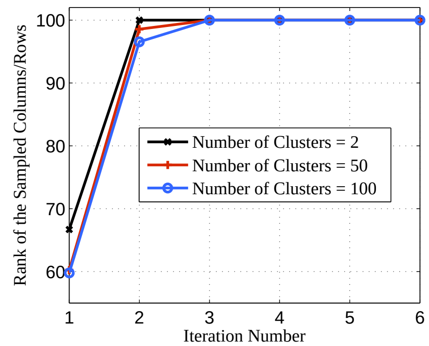

Fig. 10. The rank of the matrix of sampled columns.

E. Online Implementation

In this section, the proposed online method is examined. It is shown that the proposed scalable online algorithm tracks the underlying subspace successfully. The matrix $\mathbf{S}$ follows the Bernoulli model with $\rho = 0.01$. Assume that the orthonormal matrix $\mathbf{U} \in \mathbb{R}^{N_1 \times r}$ spans a random r-dimensional subspace. The matrix $\mathbf{L}$ is generated as follows.

For k from 1 to N_2
 1. Generate $\mathbf{E} \in \mathbb{R}^{N_1 \times r}$ and $\mathbf{q} \in \mathbb{R}^{r \times 1}$ randomly.
 2. $\mathbf{L} = [\mathbf{L} \;\; \mathbf{U}\mathbf{q}]$.
 3. If $(\text{mod}(k, n) = 0)$
  $\mathbf{U} = \text{approx-r}(\mathbf{U} + \alpha \mathbf{E})$.
 End If
End For

The elements of $\mathbf{q}_i$ and $\mathbf{E}$ are sampled from standard normal distributions. The output of the function approx-r is the matrix of the first r left singular vectors of the input matrix and $\text{mod}(k, n)$ is the remainder of k/n. The parameters α and n control the rate of change of the underlying subspace. The subspace changes at a higher rate if α is increased or n is decreased. In this simulation, $n = 10$, i.e., the CS is randomly rotated every 10 new data columns. In this simulation, the parameter $r = 5$ and $N_1 = 400$. We compare the performance of the proposed online approach to the online algorithm in [35]. For our proposed method, we set $C = 20$ when Algorithm 2 is applied to $\mathbf{U}$, i.e., $20r$ rows of $\mathbf{U}$ are sampled. The method presented in [35] is initialized with the exact CS and its tuning parameter is set equal to $1/\sqrt{N_1}$. The algorithm [35] updates the CS with every new data column. The parameter n_u of the proposed online method is set equal to 4 (i.e., the CS is updated every 4 new data columns) and the parameter n_s is set equal to $5r$. Define $\hat{\mathbf{L}}$ as the recovered LR matrix. Fig. 11 shows the ℓ_2-norm of the columns of

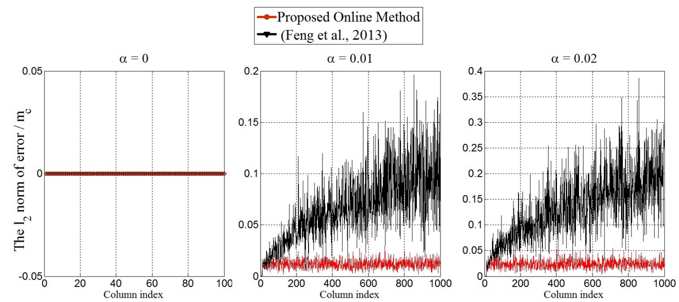

Fig. 11. Performance of the proposed online approach and the online algorithm in [35].

$\mathbf{L} - \hat{\mathbf{L}}$ normalized by the average ℓ_2-norm of the columns of $\mathbf{L}$ for different values of α. One can see that the proposed method can successfully track the CS while it is continuously changing. The online method [35] performs well when the subspace is not changing ($\alpha = 0$), however, it fails to track the subspace when it is changing.

Appendix

Proof of Lemma 2

The selected columns of $\mathbf{L}$ can be written as $\mathbf{L}_{s1} = \mathbf{L}\mathbf{S}_1$. Using the compact SVD of $\mathbf{L}$, $\mathbf{L}_{s1}$ can be rewritten as $\mathbf{L}_{s1} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{S}_1$. Therefore, to show that the CS of $\mathbf{L}_{s1}$ is equal to that of $\mathbf{L}$, it suffices to show that the matrix $\mathbf{V}^T\mathbf{S}_1$ is a full rank matrix. The matrix $\mathbf{S}_1$ selects m_1 rows of $\mathbf{V}$ uniformly at random. Therefore, using Theorem 2 in [11], if

$$m_1 \geq r\gamma^2(\mathbf{V}) \max\left(c_2 \log r, c_3 \log \frac{3}{\delta}\right), \tag{25}$$

then the matrix $\mathbf{V}^T\mathbf{S}_1$ satisfies the inequality

$$\|I - \frac{N_2}{m_1}\mathbf{V}^T\mathbf{S}_1\mathbf{S}_1^T\mathbf{V}\| \leq \frac{1}{2} \tag{26}$$

with probability at least $(1-\delta)$, where c_2, c_3 are numerical constants [11]. Accordingly, if σ_1 and σ_r denote the largest and smallest singular values of $\mathbf{S}_1^T\mathbf{V}$, respectively, then

$$\frac{m_1}{2N_2} \leq \sigma_1^2 \leq \sigma_r^2 \leq \frac{3m_1}{2N_2} \tag{27}$$

Therefore, the singular values of the matrix $\mathbf{V}^T\mathbf{S}_1$ are greater than $\sqrt{\frac{m_1}{2N_2}}$. Accordingly, the matrix $\mathbf{V}^T\mathbf{S}_1$ is a full rank matrix.

Remark 3. *A direct application of Theorem 2 in [11] would in fact lead to the sufficient condition*

$$m_1 \geq r\gamma^2(\mathbf{R}) \max\left(c_2 \log r, c_3 \log \frac{3}{\delta}\right), \tag{28}$$

where $\mathbf{R} \in \mathbb{R}^{N_2 \times N_2}$ *denotes the matrix of right singular vectors of* $\mathbf{L}$. *The bound in (25) is slightly tighter since it uses the incoherence parameter* $\gamma(\mathbf{V}) \leq \gamma(\mathbf{R}) \triangleq \sqrt{N_2}\max_{i,j}|\mathbf{R}(i,j)|$ *in (28), where* $\mathbf{V}$ *consists of the first* r *columns of* $\mathbf{R}$. *This follows easily by replacing the incoherence parameter in the step that bounds the* ℓ_2*-norm of the row vectors of the submatrix in the proof of ([11], Theorem 2).*

Proof of lemma 3

The sampled columns are written as

$$\mathbf{D}_{s1} = \mathbf{D}\mathbf{S}_1 = \mathbf{L}_{s1} + \mathbf{S}_{s1}. \tag{29}$$

First, we investigate the coherency of the new LR matrix $\mathbf{L}_{s1}$. Define $\mathbf{P}_{\mathbf{S}_1^T\mathbf{V}}$ as the projection matrix onto the CS of $\mathbf{S}_1^T\mathbf{V}$ which is equal to the rows subspace of $\mathbf{L}_{s1}$. Therefore, the projection of the standard basis onto the rows subspace of $\mathbf{L}_{s1}$ can be written as

$$\begin{aligned}
&\max_i \|\mathbf{P}_{\mathbf{S}_1^T\mathbf{V}}\mathbf{e}_i\|_2^2 = \max_i \|\mathbf{S}_1^T\mathbf{V}(\mathbf{V}^T\mathbf{S}_1\mathbf{S}_1^T\mathbf{V})^{-1}\mathbf{V}^T\mathbf{S}_1\mathbf{e}_i\|_2^2 \\
&\leq \max_j \|\mathbf{S}_1^T\mathbf{V}(\mathbf{V}^T\mathbf{S}_1\mathbf{S}_1^T\mathbf{V})^{-1}\mathbf{V}^T\mathbf{e}_j\|_2^2 \\
&\leq \|\mathbf{S}_1^T\mathbf{V}(\mathbf{V}^T\mathbf{S}_1\mathbf{S}_1^T\mathbf{V})^{-1}\|^2\|\mathbf{V}^T\mathbf{e}_j\|_2^2 \\
&\leq \frac{\gamma^2(\mathbf{V})r}{N_2}(\frac{\sigma_1^2}{\sigma_r^4}) = \frac{\gamma^2(\mathbf{V})r}{N_2}\frac{6N_2}{m_1} = \frac{(6\gamma^2(\mathbf{V}))r}{m_1}
\end{aligned} \tag{30}$$

where $(\mathbf{S}_1^T\mathbf{V}(\mathbf{V}^T\mathbf{S}_1\mathbf{S}_1^T\mathbf{V})^{-1}\mathbf{V}^T\mathbf{S}_1)$ is the projection matrix onto the CS of $\mathbf{S}_1^T\mathbf{V}$. The first inequality follows from the fact that $\{\mathbf{S}_1\mathbf{e}_i\}_{i=1}^{m_1}$ is a subset of $\{\mathbf{e}_j\}_{j=1}^{N_2}$. The second inequality follows from Cauchy-Schwarz inequality and the third inequality follows from (4) and (27).

Using lemma 2.2 of [20], there exists numerical constants c_7, c_8 such that

$$\max_i \|\mathbf{U}^T e_i\|_2^2 \leq \frac{\mu_p r}{N_1} \tag{31}$$

with probability at least $1 - c_8 N_1^{-3}$ and $\mu_p = \frac{c_7 \max(r, \log N_1)}{r}$. In addition, we need to find a bound similar to the third condition of (3) for the LR matrix $\mathbf{L}_{s1}$. Let $\mathbf{L}_{s1} = \mathbf{U}_{s1}\boldsymbol{\Sigma}_{s1}\mathbf{V}_{s1}^T$ be the SVD decomposition of $\mathbf{L}_{s1}$. Define

$$\mathbf{H} = \mathbf{U}_{s1}\mathbf{V}_{s1}^T = \sum_{i=1}^{r}\mathbf{U}_{s1}^i(\mathbf{V}_{s1}^i)^T \tag{32}$$

where $\mathbf{U}_{s1}^i$ is the i^{th} column of $\mathbf{U}_{s1}$ and $\mathbf{V}_{s1}^i$ is the i^{th} column of $\mathbf{V}_{s1}$. Given the random orthogonal model of the CS, $\mathbf{H}$ has the same distribution as

$$\mathbf{H}^{'} = \sum_{i=1}^{r}\epsilon_i\mathbf{U}_{s1}^i(\mathbf{V}_{s1}^i)^T \tag{33}$$

where $\{\epsilon_i\}$ is an independent Rademacher sequence. Using Hoeffding's inequality [36], conditioned on $\mathbf{U}_{s1}$ and $\mathbf{V}_{s1}$ we have

$$\begin{aligned}
&\mathbb{P}\left(|\mathbf{H}^{'}(i,j)| > t\right) \leq 2e^{\frac{-t^2}{2h_{ij}^2}}, \\
&h_{ij}^2 = \sum_{k=1}^{r}(\mathbf{U}_{s1}(i,k))^2(\mathbf{V}_{s1}(j,k))^2
\end{aligned} \tag{34}$$

Consider the following lemma adapted from Lemma 2.2 of [20].

Lemma 6 (Adapted from lemma 2.2 of [20])**.** *If the orthonormal matrix* $\mathbf{U}$ *follows the random orthogonal model, then*

$$\mathbb{P}\left(|\mathbf{U}(i,j)|^2 \geq 20\frac{\log N_1}{N_1}\right) \leq 3N_1^{-8}. \tag{35}$$

Therefore,

$$|\mathbf{U}_{s1}(i,k)|^2 \leq 20\frac{\log N_1}{N_1} \tag{36}$$

with probability at least $1-3N_1^{-8}$. Therefore, we can bound h_{ij}^2 as

$$h_{ij}^2 \leq 20\frac{\log N_1}{N_1}\|\mathbf{V}_{s1}e_i\|_2^2 \tag{37}$$

Using (30), (37) can be rewritten as

$$h_{ij}^2 \leq 120\frac{\log N_1 \gamma^2(\mathbf{V})r}{N_1 m_1} \tag{38}$$

Choose $t = \omega\frac{\gamma(\mathbf{V})\sqrt{r}}{\sqrt{N_1 m_1}}$ for some constant ω. Thus, the unconditional form of (34) can be written as

$$\begin{aligned}&\mathbb{P}\left(|\mathbf{H}^{'}(i,j)| > \omega\frac{\gamma(\mathbf{V})\sqrt{r}}{\sqrt{N_1 m_1}}\right) \leq 2e^{\frac{-\zeta\omega^2}{\log N_1}} + \\ &\mathbb{P}\left(h_{ij}^2 \geq 120\frac{\log N_1\gamma^2(\mathbf{V})r}{N_1 m_1}\right)\end{aligned} \tag{39}$$

for some numerical constant ζ. Setting $\omega = \zeta^{'}\log N_1$ where $\zeta^{'}$ is a sufficiently large numerical constant gives

$$\mathbb{P}\left(\|\mathbf{H}^{'}\|_\infty \geq c_9 \log N_1 \frac{\gamma(\mathbf{V})\sqrt{r}}{\sqrt{N_1 m_1}}\right) \leq 3rN_1^{-7} \tag{40}$$

for come constant number c_9 since (36) should be satisfied for rN_1 random variables.

Therefore, according to lemma 1, if

$$m_1 \geq \frac{r}{\rho_r}\mu^{'}(\log N_1)^2 \tag{41}$$

$$\rho \leq \rho_s \tag{42}$$

then, the convex algorithm (9) yields the exact decomposition with probability at least $1-c_8N_1^{-3}$ where

$$\mu^{'} = \max\left(\mu_p, 6\gamma^2(\mathbf{V}), (\gamma(\mathbf{V})c_9\log N_1)^2\right). \tag{43}$$

Proof of lemma 4

Based on (1), the matrix of sampled rows can be written as

$$\mathbf{D}_{s2} = \mathbf{S}_2^T\mathbf{D} = \mathbf{S}_2^T\mathbf{L} + \mathbf{S}_2^T\mathbf{S} = \mathbf{L}_{s2} + \mathbf{S}_{s2} \tag{44}$$

Let $\mathbf{L}_{s2} = \mathbf{U}_{s2}\mathbf{\Sigma}_{s2}\mathbf{V}_{s2}^T$ be the compact SVD decomposition of $\mathbf{L}_{s2}$ and $\mathbf{L}_{s2} = \mathbf{U}_{s2}^c\mathbf{\Sigma}_{s2}^c(\mathbf{V}_{s2}^c)^T$ its complete SVD. It can be shown [8] that (13) is equivalent to

$$\begin{aligned}&\min_{\hat{\mathbf{z}}_i} && \|\hat{\mathbf{z}}_i\|_1 \\ &\text{subject to} && (\mathbf{U}_{s2}^{\perp})^T\hat{\mathbf{z}}_i = (\mathbf{U}_{s2}^{\perp})^T\mathbf{S}_{s2}^i\,.\end{aligned} \tag{45}$$

where $\mathbf{S}_{s2}^i$ is the i^{th} column of $\mathbf{S}_{s2}$ and $\mathbf{U}_{s2}^{\perp}$ is the last (m_2-r) columns of $\mathbf{U}_{s2}^c$ which are orthogonal to $\mathbf{U}_{s2}$. In other words, if $\mathbf{q}_i^*$ is the optimal point of (13) and $\mathbf{z}_i^* \in \mathbb{R}^{m_2}$ is the optimal point of (45), then $\mathbf{z}_i^* = \mathbf{S}_2^T(\mathbf{d}_i - \mathbf{U}\mathbf{q}_i^*)$. Thus, it is enough to show that the optimal point of (45) is equal to $\mathbf{S}_{s2}^i$.

The columns subspace of $\mathbf{U}_{s2}$ obeys the random orthogonal model. Thus, $\mathbf{U}_{s2}^{\perp}$ can be modeled as a random subset of $\mathbf{U}_{s2}^c$. Based on the result in [11], if we assume that the sign of the non-zero elements of $\mathbf{S}_{s2}^i$ are uniformly random, then the optimal point of (45) is $\mathbf{S}_{s2}^i$ with probability at least $(1-\delta)$ provided that

$$m_2 - r \geq \max\left(c_4\|\mathbf{S}_{s2}^i\|_0\gamma^2(\mathbf{U}_{s2}^c)\log\frac{m_2}{\delta}, c_5\left(\log\frac{m_2}{\delta}\right)^2\right) \tag{46}$$

for some fixed numerical constants c_4 and c_5. The parameter $\gamma(\mathbf{U}_{s2}^c) = \sqrt{m_2}\max_{i,j}|\mathbf{U}_{s2}^c(i,j)|$ and $\|\mathbf{S}_{s2}^i\|_0$ is the l_0-norm of $\mathbf{S}_{s2}^i$. In this paper, we do not assume that the sign of the non-zero elements of the sparse matrix $\mathbf{S}$ is random. However, according to Theorem 2.3 of [1] (de-randomization technique) if the locations of the nonzero entries of $\mathbf{S}$ follow the Bernoulli model with parameter 2ρ, and the signs of $\mathbf{S}$ are uniformly random and if (45) yields the exact solution who, then it is also exact with at least the same probability for the model in which the signs are fixed and the locations follow the Bernoulli model with parameter ρ [1]. Therefore, it suffices to provide the sufficient condition for the exact recovery of a random sign sparse vector with Bernoulli parameter 2ρ.

First, we provide sufficient conditions to guarantee that

$$m_2 - r \geq c_4\|\mathbf{S}_{s2}^i\|_0\gamma^2(\mathbf{U}_{s2}^c)\log\frac{m_2}{\delta} \tag{47}$$

with high probability. Using Lemma 6 and the union bound,

$$\max_{i,j}|\mathbf{U}_{s2}^c(i,j)|^2 \leq 20\frac{\log m_2}{m_2} \tag{48}$$

with probability at least $1-3m_2^{-6}$.

Now, we find the sufficient number of randomly sampled rows, m_2, to guarantee that (47) is satisfied with high probability. It is obvious that $m_2 < N_1$. Define $\kappa = \frac{\log N_1}{r}$. Therefore, it is sufficient to show that

$$\frac{m_2}{\|\mathbf{S}_{s2}^i\|_0} \geq r\left(c_6\kappa\log\frac{N_1}{\delta} + 1\right) \tag{49}$$

whp, where $c_6 = 20c_4$. Suppose that

$$\rho \leq \frac{1}{\beta r\left(c_6\kappa\log\frac{N_1}{\delta} + 1\right)} \tag{50}$$

where β is a real number greater than one. Define $\alpha = r\left(c_6\kappa\log\frac{N_1}{\delta} + 1\right)$. According to (50) and the Chernoff Bound for Binomial random variables [37], we have

$$\mathbb{P}\left(\|\mathbf{S}_{s2}^i\|_0 - \frac{m_2}{\beta\alpha} > a\right) \leq \exp\left(\frac{-a^2}{2(\frac{m_2}{\alpha\beta} + \frac{a}{3})}\right). \tag{51}$$

If we set $a = \frac{m_2}{\alpha}\left(1 - \frac{1}{\beta}\right)$, then the inequality (49) is satisfied. Therefore, (51) can be rewritten as

$$\begin{aligned}&\mathbb{P}\left(\|\mathbf{S}_{s2}^i\|_0 - \frac{m_2}{\beta\alpha} > \frac{m_2}{\alpha}\left(1-\frac{1}{\beta}\right)\right) \\ &\leq 2\exp\left(\frac{-m_2^2(\beta-1)^2}{\alpha^2\beta^2}\frac{3\alpha\beta}{2m_2(\beta+2)}\right).\end{aligned} \tag{52}$$

Therefore, if

$$m_2 \geq \frac{2r\beta(\beta-2)\log\left(\frac{1}{\delta}\right)}{3(\beta-1)^2}\left(c_6\kappa\log\frac{N_1}{\delta} + 1\right), \tag{53}$$

then the inequality (49) is satisfied with probability at least $(1-\delta)$. Accordingly, if

$$\rho \leq \frac{0.5}{r\beta\left(c_6\kappa\log\frac{N_1}{\delta}+1\right)}$$
$$m_2 \geq \max\left(\frac{2r\beta(\beta-2)\log\left(\frac{1}{\delta}\right)}{3(\beta-1)^2}\left(c_6\kappa\log\frac{N_1}{\delta}+1\right),\ \sqrt[6]{\frac{3}{\delta}},\ c_5\left(\log\frac{N_1}{\delta}\right)^2\right) \tag{54}$$

then (45) returns the exact sparse vector with probability at least $1-3\delta$. The factor 0.5 in the numerator of the right hand side of the first inequality of (54) is due to the derandomization technique [1] to provide the guarantee for the fixed sign case.

Proof of Theorem 5

The proposed decomposition algorithm yields the exact decomposition if:

1. The sampled columns of the LR matrix span the the columns subspace of $\mathbf{L}$. Lemma 2 provides the sufficient conditions on m_1 to guarantee that the columns of $\mathbf{L}_{s1}$ span $\mathcal{U}$ with high probability.

2. The program (9) yields the correct LR and sparse components of $\mathbf{D}_{s1}$. Lemma 3 provides the sufficient conditions on m_1 and ρ to guarantee that $\mathbf{D}_{s1}$ is decomposed correctly with high probability.

3. The sampled rows of the LR matrix span the rows subspace of $\mathbf{L}$. Since it is assumed that the CS of $\mathbf{L}$ is sampled from the random orthogonal model, according to Lemma 6

$$\mathbb{P}\left(\max_{i,j}|\mathbf{U}(i,j)|^2 \geq 20\frac{\log N_1}{N_1}\right) \leq 3rN_1^{-7}\,. \tag{55}$$

Therefore, according to Lemma 2 if

$$m_2 \geq r\log N_1 \max\left(c_2^{'}\log r, c_3^{'}\log\left(\frac{3}{\delta}\right)\right), \tag{56}$$

then the selected rows of the matrix $\mathbf{L}$ span the rows subspace of $\mathbf{L}$ with probability at least $(1-\delta-3rN_1^{-7})$ where $c_2^{'}$ and $c_3^{'}$ are numerical constants.

4. The minimization (15) yields the correct RS. Lemma 4 provides the sufficient conditions to ensure that (13) yields the correct representation vector. In order to guarantee the performance of (15), we substitute δ with δ/N_2 since (15) has to return exact representation for the columns of $\mathbf{D}$. Therefore,

$$\mathbb{P}\left(\text{Incorrect Decomposition}\right) \leq \delta + c_8 N_1^{-3} + \delta + 3rN_1^{-7} + 3\delta.$$

References

[1] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM (JACM)*, vol. 58, no. 3, p. 11, 2011.

[2] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky, "Rank-sparsity incoherence for matrix decomposition," *SIAM Journal on Optimization*, vol. 21, no. 2, pp. 572–596, 2011.

[3] X. Yuan and J. Yang, "Sparse and low-rank matrix decomposition via alternating direction methods," *preprint*, vol. 12, 2009.

[4] Z. Lin, M. Chen, and Y. Ma, "The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[5] L. W. Mackey, M. I. Jordan, and A. Talwalkar, "Divide-and-conquer matrix factorization," in *Advances in Neural Information Processing Systems (NIPS)*, 2011, pp. 1134–1142.

[6] L. Mackey, A. Talwalkar, and M. I. Jordan, "Distributed matrix completion and robust factorization," *arXiv preprint arXiv:1107.0789*, 2011.

[7] X. Li and J. Haupt, "Identifying outliers in large matrices via randomized adaptive compressive sampling," *IEEE Transactions on Signal Processing*, vol. 63, no. 7, pp. 1792–1807, 2015.

[8] E. J. Candès and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[9] S. Minaee and Y. Wang, "Screen content image segmentation using least absolute deviation fitting," in *Image Processing (ICIP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 3295–3299.

[10] S. Minaee, A. Abdolrashidi, and Y. Wang, "Screen content image segmentation using sparse-smooth decomposition," *arXiv preprint arXiv:1511.06911*, 2015.

[11] E. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse problems*, vol. 23, no. 3, p. 969, 2007.

[12] J. Wright, A. Ganesh, K. Min, and Y. Ma, "Compressive principal component pursuit," *Information and Inference*, vol. 2, no. 1, pp. 32–68, 2013.

[13] M. Rahmani and G. Atia, "Randomized robust subspace recovery for high dimensional data matrices," *arXiv preprint arXiv:1505.05901*, 2015.

[14] M. Rahmani and G. K. Atia, "Analysis of randomized robust pca for high dimensional data," in *Signal Processing and Signal Processing Education Workshop (SP/SPE), 2015 IEEE*. IEEE, 2015, pp. 25–30.

[15] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM review*, vol. 53, no. 2, pp. 217–288, 2011.

[16] T. Zhou and D. Tao, "Godec: Randomized low-rank & sparse matrix decomposition in noisy case," in *International conference on machine learning (ICML)*. Omnipress, 2011.

[17] Y. Mu, J. Dong, X. Yuan, and S. Yan, "Accelerated low-rank visual recovery by random projection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011, pp. 2609–2616.

[18] R. Liu, Z. Lin, S. Wei, and Z. Su, "Solving principal component pursuit in linear time via l_1 filtering," *arXiv preprint arXiv:1108.5359*, 2011.

[19] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, "A theory of pseudoskeleton approximations," *Linear Algebra and its Applications*, vol. 261, no. 1, pp. 1–21, 1997.

[20] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[21] C. Boutsidis, M. W. Mahoney, and P. Drineas, "An improved approximation algorithm for the column subset selection problem," in *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 2009, pp. 968–977.

[22] M. W. Mahoney, "Randomized algorithms for matrices and data," *Foundations and Trends® in Machine Learning*, vol. 3, no. 2, pp. 123–224, 2011.

[23] G. Liu and P. Li, "Recovery of coherent data via low-rank dictionary pursuit," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 1206–1214.

[24] M. Rahmani and G. Atia, "Innovation pursuit: A new approach to subspace clustering," *arXiv preprint arXiv:1512.00907*, 2015.

[25] M. Joneidi, P. Ahmadi, M. Sadeghi, and N. Rahnavard, "Union of low-rank subspaces detector," *Signal Processing, IET*, vol. 10, no. 1, pp. 55–62, 2016.

[26] M. Rudelson and R. Vershynin, "Sampling from large matrices: An approach through geometric functional analysis," *Journal of the ACM (JACM)*, vol. 54, no. 4, p. 21, 2007.

[27] A. Deshpande and L. Rademacher, "Efficient volume sampling for row/column subset selection," in *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2010, pp. 329–338.

[28] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[29] Q. Ke and T. Kanade, "Robust l 1 norm factorization in the presence of outliers and missing data by alternative convex programming," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2005, pp. 739–746.

[30] Z. Zhou, X. Li, J. Wright, E. Candès, and Y. Ma, "Stable principal component pursuit," in *IEEE International Symposium on Information Theory Proceedings (ISIT)*. IEEE, 2010, pp. 1518–1522.

[31] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[32] E. Candès and J. Romberg, "l1-magic: Recovery of sparse signals via convex programming," *URL: www. acm. caltech. edu/l1magic/downloads/l1magic. pdf*, vol. 4, p. 14, 2005.

[33] T. Bouwmans, A. Sobral, S. Javed, S. K. Jung, and E.-H. Zahzah, "Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset," *arXiv preprint arXiv:1511.01245*, 2015.

[34] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian, "Statistical modeling of complex backgrounds for foreground object detection," *IEEE Transactions on Image Processing*, vol. 13, no. 11, pp. 1459–1472, 2004.

[35] J. Feng, H. Xu, and S. Yan, "Online robust PCA via stochastic optimization," in *Advances in Neural Information Processing Systems (NIPS)*, 2013, pp. 404–412.

[36] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American statistical association*, vol. 58, no. 301, pp. 13–30, 1963.

[37] C. McDiarmid, "Concentration," in *Probabilistic methods for algorithmic discrete mathematics*. Springer, 1998, pp. 195–248.